	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: García Morales Karina

Asignatura: Fundamentos de la Programación

Grupo: 22

No. de práctica(s): 2

Integrante(s): Ian Santiago Rivera González

No. de lista o brigada: 38

Semestre: 2026-1

Fecha de entrega: 02/SEPTIEMBRE/2025

Observaciones: Ninguna

CALIFICACIÓN: _____

GNU/Linux

Objetivo:

El alumno identificará al sistema operativo como una parte esencial de un sistema de cómputo. Explorará un sistema operativo GNU/Linux con el fin de conocer y utilizar sus comandos básicos.

DESARROLLO: El Sistema Operativo es el conjunto de programas y datos que administra los recursos tanto de hardware como de software de un sistema de cómputo. Los componentes de un sistema operativo, en general, son:

•Gestor de memoria,

• Administrador y planificador de procesos,

• Sistema de archivos y

• Administración de E/S.

Se encuentran en el kernel o núcleo del sistema operativo.

En la Interfaz con el usuario, las hay de tipo texto y de tipo gráfico. Actualmente se usan mas las de tipo grafico como en los celulares, por ejemplo. Pero al desarrollar proyectos donde se usan documentos y programas se hace necesario usar dispositivos de entrada y salida y aplicaciones en modo de texto.

Sistema Operativo Linux

Es un sistema operativo tipo Unix de libre distribución para computadoras personales, servidores y estaciones de trabajo.

Está conformado por el núcleo (kernel) y varios programas y bibliotecas. Muchos programas y bibliotecas han sido posibles gracias al proyecto GNU, por ello, se conoce a este sistema operativo como GNU/Linux.

Software libre

Es el que se puede adquirir de manera gratuita, es decir, no se tiene que pagar algún tipo de licencia a alguna casa desarrolladora de software por el uso de éste.

Que un software sea libre implica también que el software viene acompañado del código fuente, ósea que se pueden realizar cambios en el funcionamiento del sistema.

Linux se distribuye bajo la Licencia Pública General de GNU por ello, el código fuente tiene que estar siempre accesible y cualquier modificación o trabajo derivado debe tener esta licencia.

Licencia GNU

La Licencia Pública General de GNU o GNU General Public License (GNU GPL) es una licencia creada por la Free Software Foundation en 1989 y está destinada a proteger la libre distribución, modificación y uso de software.

Kernel de GNU/Linux

El kernel o núcleo de linux se define como el corazón del sistema operativo. Es el encargado de que el software y el hardware del equipo se puedan comunicar. Sus componentes son: Aplicaciones, Shell, Hardware y el sistema de archivos.

Entre el kernel y las aplicaciones existe una capa que permite al usuario comunicarse con el sistema operativo y en general con la computadora, esto a través de programas que ya vienen instalados con la distribución de Linux (Debian, Ubuntu, Fedora, etc.) y trabajan en modo gráfico o en modo texto. Uno de estos programas es el Shell.

La estructura de Linux para el almacenamiento de archivos es de forma jerárquica; por lo que la carpeta o archivo base es “root” (raíz) y se representa con una diagonal (/). De este parten todos los demás. Los archivos pueden ser carpetas (directorios), de datos, aplicaciones, programas, etc.

Interfaz de línea de comandos (CLI) o shell de GNU/Linux

El Shell de GNU/Linux permite introducir órdenes (comandos) y ejecutar programas. Todas las órdenes de UNIX/Linux son programas que están almacenados en el sistema de archivos (comandos), por ello, todo en GNU/Linux se puede controlar mediante

comandos.

Comandos básicos

Para trabajar en Linux utilizando comandos, se debe abrir una “terminal” o “consola” que es una ventana donde aparece la “línea de comandos” en la cual se escribirá la orden o comando. La terminal permite un mayor grado de funciones y configuración de lo que queremos hacer con una aplicación o acción en general respecto a un entorno gráfico.

La sintaxis que siguen los comandos es la siguiente:

comando [-opciones] [argumentos]

Es el nombre del comando, seguido de algunas opciones para modificar la ejecución de este y, al final, se puede incluir un argumento (ruta, ubicación, archivo, etcétera) dependiendo del comando. Tanto las opciones como los argumentos son opcionales.

RESUMEN DE LOS COMANDOS, FUNCIONES Y EJEMPLOS.

El comando ls permite listar los elementos que existen en alguna ubicación del sistema de archivos. Por defecto lista los elementos que existen en la ubicación actual; Linux nombra la ubicación actual con un punto (.), Por eso ls y ls . se pueden usar para lo mismo.

El comando *ls* realiza acciones distintas dependiendo de las banderas que utilice, por ejemplo, si se utiliza la opción l se genera un listado largo de la ubicación actual:

ls –l

Si se quiere ver los archivos que se encuentran en la raíz, usamos:

ls /

Para ver los usuarios del equipo local, revisamos el directorio *home* que parte de la raíz (/), por ejemplo:

ls /home

Tanto las opciones como los argumentos se pueden combinar para generar una ejecución más específica:

ls –l /home

El comando *man*, permite visualizar la descripción de cualquier comando, la manera en la que se puede utilizar es esta:

man ls

Si se desea ver la lista de los archivos del directorio *usr*, por ejemplo, podemos escribir el comando: ls /usr

El argumento se inicia con / indicando que es el directorio raíz, seguido de *usr* que es el nombre del directorio. Cuando especificamos la ubicación de un archivo partiendo de la raíz, se dice que estamos indicando la "ruta absoluta" del archivo.

Existe otra forma de especificar la ubicación de un archivo, esto es empleando la "ruta relativa".

Si bien el punto (.) es para indicar la ubicación actual, el doble punto (..) se utiliza para referirse al directorio "padre". De esta forma si deseamos listar los archivos que dependen del directorio padre se escribe el siguiente comando:

ls ..

o

ls ../

Se pueden utilizar varias referencias al directorio padre para ir navegando por el sistema de archivos, de tal manera que se realice la ubicación de un archivo a través de una ruta relativa. Si nuestra cuenta depende de *home*, la ruta relativa para listar los archivos del directorio *usr* es:

ls ../../usr

Con los primeros dos puntos se hace referencia al directorio home, con los siguientes dos puntos se refiere al directorio raíz, y finalmente se escribe el nombre del directorio *usr*.

Ejemplo (comando touch) El comando *touch* permite crear un archivo de texto, su sintaxis es la siguiente:

touch nombre_archivo[.ext]

En GNU/Linux no es necesario agregar una extensión al archivo creado, sin embargo, es recomendable hacerlo para poder identificar el tipo de archivo creado.

Ejemplo (comando mkdir) El comando *mkdir* permite crear una carpeta, su sintaxis es la siguiente:

mkdir nombre_carpeta

Ejemplo (comando cd) El comando *cd* permite ubicarse en una carpeta, su sintaxis es la siguiente:

cd nombre_carpeta

Si deseamos situarnos en la carpeta de inicio, la carpeta padre, escribimos el comando:

cd ..

Ejemplo (comando pwd) El comando *pwd* permite conocer la ubicación actual(ruta), su sintaxis es la siguiente: pwd

Ejemplo (comando find) El comando *find* permite buscar un elemento dentro del sistema de archivos, su sintaxis es la siguiente: find . –name cadena_buscar

Al comando *find* hay que indicarle en qué parte del sistema de archivos va a iniciar la búsqueda. Utilizando la bandera –name permite determinar la cadena a buscar (comúnmente es el nombre de un archivo).

Si queremos encontrar la ubicación del archivo *tareas*, por ejemplo se escribe el siguiente comando: find . –name tareas

Ejemplo (comando clear) El comando *clear* permite limpiar la consola o terminal, su sintaxis es la siguiente: clear

Ejemplo (comando cp) El comando *cp* permite copiar un archivo, su sintaxis es la siguiente: cp archivo_origen archivo_destino

Si queremos una copia del archivo *datos.txt* con nombre *datosViejos.txt* en el mismo directorio, entonces se escribe el comando

cp datos.txt datosViejos.txt

Ahora, si requerimos una copia de un archivo que está en la carpeta padre en la ubicación actual y con el mismo nombre, entonces podemos emplear las rutas relativas de la siguiente forma:

cp ../archivo_a_copiar .

Es muy importante indicar como archivo destino al punto (.) para que el archivo de copia se ubique en el directorio actual.

Ejemplo (comando mv) El comando *mv* mueve un archivo de un lugar a otro, en el sistema de archivos; su sintaxis es la siguiente:

mv ubicación_origen/archivo ubicación_destino

El comando mueve el archivo desde su ubicación origen hacia la ubicación deseada(destino).

Si queremos que un archivo que está en la carpeta padre, reubicarlo en el directorio actual y con el mismo nombre, entonces podemos emplear las rutas relativas de la siguiente forma:

mv ../archivo_a_reubicar .

Este comando también puede ser usado para cambiar el nombre de un archivo, simplemente se indica el nombre actual del archivo y el nuevo nombre:

mv nombre_actual_archivo nombre_nuevo_archivo

Ejemplo (comando rm) El comando *rm* permite eliminar un archivo o un directorio, su sintaxis es la siguiente:

rm nombre_archivo

rm nombre_carpeta

Cuando la carpeta que se desea borrar contiene información, se debe utilizar la bandera –f para forzar la eliminación. Si la carpeta contiene otras carpetas, se debe utilizar la opción –r, para realizar la eliminación recursiva.

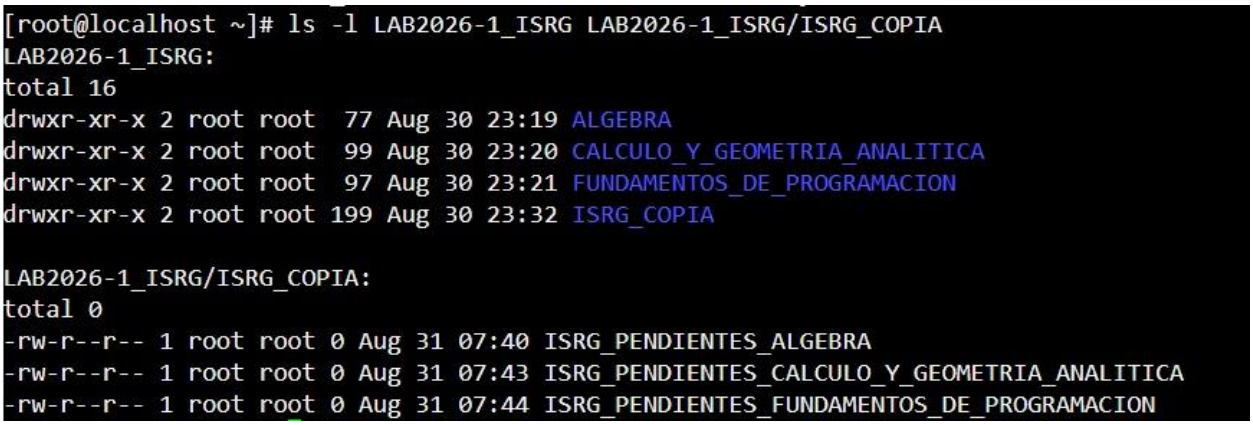
TAREA

```
Loading...

Welcome to Fedora 33 (riscv64)

[root@localhost ~]# mkdir LAB2026-1_ISRG
[root@localhost ~]# cd LAB2026-1_ISRG
[root@localhost LAB2026-1_ISRG]# mkdir ALGEBRA
[root@localhost LAB2026-1_ISRG]# mkdir CALCULO_Y_GEOMETRIA_ANALITICA
[root@localhost LAB2026-1_ISRG]# mkdir FUNDAMENTOS_DE_PROGRAMACION
[root@localhost LAB2026-1_ISRG]# cd ALGEBRA
[root@localhost ALGEBRA]# ISRG_PENDIENTES_ALGEBRA
sh: ISRG_PENDIENTES_ALGEBRA: command not found
[root@localhost ALGEBRA]# touch ISRG_PENDIENTES_ALGEBRA
[root@localhost ALGEBRA]# cd ..
[root@localhost LAB2026-1_ISRG]# cd CALCULO_Y_GEOMETRIA_ANALITICA
[root@localhost CALCULO_Y_GEOMETRIA_ANALITICA]# touch ISRG_PENDIENTES_CALCULO_Y_
GEOMETRIA_ANALITICA
[root@localhost CALCULO_Y_GEOMETRIA_ANALITICA]# cd ..
[root@localhost LAB2026-1_ISRG]# cd FUNDAMENTOS_DE_PROGRAMACION
[root@localhost FUNDAMENTOS_DE_PROGRAMACION]# touch ISRG_PENDIENTES_FUNDAMENTOS_
DE_PROGRAMACION
[root@localhost FUNDAMENTOS_DE_PROGRAMACION]# cd ..
[root@localhost LAB2026-1_ISRG]# ls
ALGEBRA  CALCULO_Y_GEOMETRIA_ANALITICA  FUNDAMENTOS_DE_PROGRAMACION
[root@localhost LAB2026-1_ISRG]# cd ALGEBRA
[root@localhost ALGEBRA]# ls
ISRG_PENDIENTES_ALGEBRA
[root@localhost ALGEBRA]# cd ..
[root@localhost LAB2026-1_ISRG]# cd CALCULO_Y_GEOMETRIA_ANALITICA
[root@localhost CALCULO_Y_GEOMETRIA_ANALITICA]# ls
ISRG_PENDIENTES_CALCULO_Y_GEOMETRIA_ANALITICA
[root@localhost CALCULO_Y_GEOMETRIA_ANALITICA]# cd ..
[root@localhost LAB2026-1_ISRG]# cd FUNDAMENTOS_DE_PROGRAMACION
[root@localhost FUNDAMENTOS_DE_PROGRAMACION]# ls
ISRG_PENDIENTES_FUNDAMENTOS_DE_PROGRAMACION
[root@localhost FUNDAMENTOS_DE_PROGRAMACION]# cd ..
[root@localhost LAB2026-1_ISRG]# cd ..
[root@localhost ~]# mkdir ISRG_COPIA
```

Del punto 1. al 7.

```
[root@localhost ~]# cp LAB2026-1_ISRG/ALGEBRA/ISRG_PENDIENTES_ALGEBRA ISRG_COPIA/
[root@localhost ~]# cp LAB2026-1_ISRG/CALCULO_Y_GEOMETRIA_ANALITICA/ISRG_PENDIENTES_CALCULO_Y_GEOMETRIA_ANALI
TICA ISRG_COPIA/
[root@localhost ~]# cp LAB2026-1_ISRG/FUNDAMENTOS_DE_PROGRAMACION/ISRG_PENDIENTES_FUNDAMENTOS_DE_PROGRAMACION
 ISRG_COPIA/
[root@localhost ~]# cd ISRG_COPIA
[root@localhost ISRG_COPIA]# ls
ISRG_PENDIENTES_ALGEBRA                      ISRG_PENDIENTES_FUNDAMENTOS_DE_PROGRAMACION
ISRG_PENDIENTES_CALCULO_Y_GEOMETRIA_ANALITICA
[root@localhost ISRG_COPIA]# cd ..
[root@localhost ~]# mv ISRG_COPIA LAB2026-1_ISRG/
```

Del punto 8. Al 10.



Punto 11.

chmod sirve para cambiar los permisos de lectura, escritura y ejecución de un archivo o directorio, mientras que **chown** sirve para cambiar el propietario y el grupo al que pertenece un archivo o directorio en sistemas Linux/Unix. Ambos comandos se complementan para gestionar el acceso a los archivos, controlando quién puede hacer qué (permisos) y quién es responsable de ellos (propietario).

Los permisos en Linux son reglas que controlan el acceso a archivos y directorios, y consisten en lectura (r), escritura (w) y ejecución (x), aplicados a tres categorías de usuarios: propietario (u), grupo (g) y otros (o).

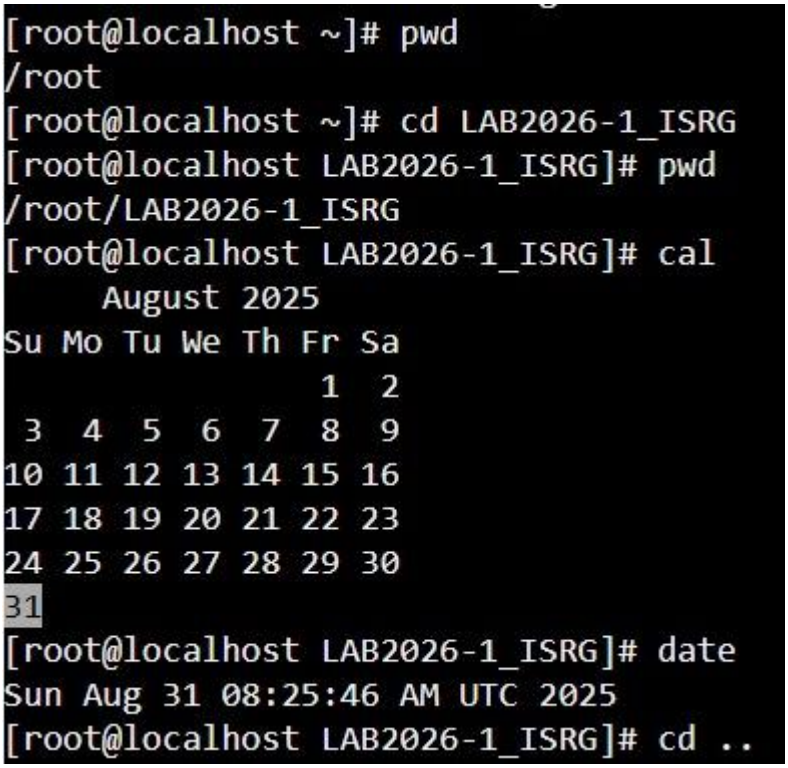
Tipos:

Lectura (r): Permite ver el contenido de un archivo o listar los archivos de un directorio.

Escritura (w): Permite modificar el contenido de un archivo o crear/eliminar archivos dentro de un directorio.

Ejecución (x): Permite ejecutar un archivo o entrar a un directorio.

En el caso del directorio del laboratorio, por ejemplo, rwxr-xr-x es un permiso común para directorios donde el propietario tiene control total, y el grupo y otros pueden leer y ejecutar y hay una d al principio porque quiere decir directorio; -rw-r--r—nos dice al inicio con el guion que es un archivo y no un directorio, pero que además solo el usuario tiene los permisos de leer y escribir; el grupo y los demás solo pueden leer.



Del punto 12. al 14.

Al escribir el comando “cal” me arrojó un calendario del mes y año en el que nos encontramos, en este caso agosto de 2025, además me remarcó el día en el que estaba que fue el 31 de agosto.

Al escribir el comando “date” me dio la fecha en la que estaba el día que realicé la tarea que fue el 31 de agosto de 2025, pero también, me dio la hora.

Del punto 15. al 16.

El comando *man*, permite visualizar la descripción de cualquier comando y se usa para ver la función de los comandos.

El comando cat, su nombre deriva de la palabra concatenar y te permite crear, fusionar o imprimir archivos en la pantalla de salida estándar o en otro archivo y mucho más.

Conclusión

Puedo concluir, que esta práctica me enseño mucho acerca del Sistema operativo de Linux, además de que pude aprender las funciones de muchísimos comandos, meparece esto me acerca un poco más al mundo de la programación y me ayuda a comprender

cuál es la esencia en el hardware y software en los sistemas computacionales. También, me mostró un poco el funcionamiento dentro de los dispositivos electrónicos, es decir, no solamente conocer el dispositivo mientras lo usas, sino realmente conocer cómo está estructurado. Finalmente, en particular esta práctica, fue bastante interesante, de momento no se me complicó demasiado, aunque, algunos comandos no comprendía cómo utilizarlos, por ejemplo, el de copiar y al principio me confundí, pero después lo pude entender.

BIBLIOGRAFÍA

Facultad de Ingeniería, UNAM. (2025). *Manual de prácticas del laboratorio de Fundamentos de Programación* (Versión 05, Código MADO-17). Universidad Nacional Autónoma de México. http://lcp02.fi-b.unam.mx/

Hostinger. (2025, 26 de marzo). *Comando cat de Linux: para qué sirve y ejemplos de uso*. Hostinger. Recuperado de https://www.hostinger.com/mx/tutoriales/comando-cat-linux